

МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение  
высшего образования  
«ВЛАДИВОСТОКСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»  
(ФГБОУ ВО «ВВГУ»)

ИНСТИТУТ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ И АНАЛИЗА ДАННЫХ  
КАФЕДРА ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ И СИСТЕМ

ОТЧЕТ  
ПО ЛАБОРАТОРНОЙ РАБОТЕ №6  
по дисциплине  
«Информатика и программирование»

Студент  
гр. БИС-25-3 \_\_\_\_\_ О.И. Шестопалова  
Ассистент  
преподавателя \_\_\_\_\_ М.В. Водяницкий

## Задание

Выполнить задания на Python и оформить отчет по стандартам ВВГУ.

**Задание 1.** Имеется список объектов Фонда с указанием уровня угрозы:

```
objects = [  
    ("Containment Cell A", 4),  
    ("Archive Vault", 1),  
    ("Bio Lab Sector", 3),  
    ("Observation Wing", 2)  
]
```

Используя sorted и лямбда-выражение, отсортируйте объекты по возрастанию уровня угрозы.

**Задание 2.** Дан список сотрудников Фонда с количеством проведенных смен и стоимостью одной смены:

```
staff_shifts = [  
    {"name": "Dr. Shaw", "shift_cost": 120, "shifts": 15},  
    {"name": "Agent Torres", "shift_cost": 90, "shifts": 22},  
    {"name": "Researcher Hall", "shift_cost": 150, "shifts": 10}  
]
```

Используя map и лямбда-выражение, создайте список общей стоимости работы каждого сотрудника

Затем найдите максимальную стоимость с помощью max .

**Задание 3.** Дан список персонала с уровнем допуска:

```
personnel = [  
    {"name": "Dr. Klein", "clearance": 2},  
    {"name": "Agent Brooks", "clearance": 4},  
    {"name": "Technician Reed", "clearance": 1}  
]
```

Используя map и лямбда-выражение, создайте новый список, где каждому сотруднику добавляется категория допуска:

- "Restricted" - уровень 1
- "Confidential" - уровни 2–3
- "Top Secret" - уровень 4 и выше

Результат должен быть списком словарей.

**Задание 4.** Дан список зон Фонда с указанием времени активности (в часах):

```
zones = [  
    {"zone": "Sector-12", "active_from": 8, "active_to": 18},  
    {"zone": "Deep Storage", "active_from": 0, "active_to": 24},  
    {"zone": "Research Wing", "active_from": 9, "active_to": 17}  
]
```

Используя filter и лямбда-выражение, выберите зоны, которые полностью работают в дневной период (с 8 до 18 включительно).

**Задание 5.** Фонд анализирует служебные отчеты. Некоторые отчеты содержат внешние ссылки, которые должны быть удалены перед архивированием

```
reports = [  
    {"author": "Dr. Moss", "text": "Analysis completed. Reference: http://external-archive.net"},  
    {"author": "Agent Lee", "text": "Incident resolved without escalation."},  
    {"author": "Dr. Patel", "text": "Supplementary data available at https://secure-research.org"},  
    {"author": "Supervisor Kane", "text": "No anomalies detected during inspection."},  
    {"author": "Researcher Bloom", "text": "Extended observations uploaded to http://research-notes.lab"},  
    {"author": "Agent Novak", "text": "Perimeter secured. No external interference observed."},  
    {"author": "Dr. Hargreeve", "text": "Full containment log stored at https://internal-db.scp"},  
    {"author": "Technician Moore", "text": "Routine maintenance completed successfully."},  
    {"author": "Dr. Alvarez", "text": "Cross-reference materials: http://crosslink.foundation"},  
    {"author": "Security Officer Tan", "text": "Shift completed without incidents."},  
    {"author": "Analyst Wright", "text": "Statistical model published at https://analysis-hub.org"},  
    {"author": "Dr. Kowalski", "text": "Behavioral deviations documented internally."},  
    {"author": "Agent Fischer", "text": "Additional footage archived: http://video-storage.sec"},  
    {"author": "Senior Researcher Hall", "text": "All test results verified and approved."},  
    {"author": "Operations Lead Grant", "text": "Emergency protocol draft shared via https://ops-share.scp"}]  
]
```

Используя filter и лямбда-выражение:

1. Отберите отчеты, содержащие ссылки (http или https)
2. Преобразуйте их так, чтобы вместо ссылки отображалось [ДАННЫЕ УДАЛЕНЫ].

**Задание 6.** Дан список SCP-объектов с указанием их класса содержания:

```
scp_objects = [  
    {"scp": "SCP-096", "class": "Euclid"},  
    {"scp": "SCP-173", "class": "Euclid"},  
    {"scp": "SCP-055", "class": "Keter"},  
    {"scp": "SCP-999", "class": "Safe"},  
    {"scp": "SCP-3001", "class": "Keter"}  
]
```

Используя filter и лямбда-выражение, сформируйте список SCP-объектов, которые требуют усиленных мер содержания

К объектам с усиленными мерами относятся все SCP, **класс которых не равен "Safe"**

Результат должен быть списком словарей исходного формата.

**Задание 7.** Дан список инцидентов с количеством задействованного персонала:

```
incidents = [  
    {"id": 101, "staff": 4},  
    {"id": 102, "staff": 12},  
    {"id": 103, "staff": 7},  
    {"id": 104, "staff": 20}  
]
```

Используя sorted и лямбда-выражение:

1. Отсортируйте инциденты по количеству персонала.
2. Оставьте только три наиболее ресурсоемких инцидента.

**Задание 8.** Дан список протоколов безопасности и их уровней критичности:

```
protocols = [  
    ("Lockdown", 5),  
    ("Evacuation", 4),  
    ("Data Wipe", 3),  
    ("Routine Scan", 1)  
]
```

Используя map и лямбда-выражение, создайте новый список строк вида:

```
"Protocol Lockdown - Criticality 5"
```

**Задание 9.** Имеется список смен охраны с указанием длительности (в часах):

```
shifts = [6, 12, 8, 24, 10, 4]
```

Используя filter и лямбда-выражение, выберите только те смены, которые:

- делятся не менее 8 часов
- не превышают 12 часов

**Задание 10.** Дан список сотрудников с результатами психологической оценки (от 0 до 100):

```
evaluations = [  
    {"name": "Agent Cole", "score": 78},  
    {"name": "Dr. Weiss", "score": 92},  
    {"name": "Technician Moore", "score": 61},  
    {"name": "Researcher Lin", "score": 88}  
]
```

Используя max и лямбда-выражение, определите сотрудника с наивысшей оценкой

Результатом должно быть имя сотрудника и его балл.

## Содержание

1	Выполнение работы .....	3
1.1	Задание 1 .....	3
1.2	Задание 2 .....	3
1.3	Задание 3 .....	4
1.4	Задание 4 .....	4
1.5	Задание 5 .....	5
1.6	Задание 6 .....	5
1.7	Задание 7 .....	6
1.8	Задание 8 .....	6
1.9	Задание 9 .....	7
1.10	Задание 10 .....	7

## 1 Выполнение работы

### 1.1 Задание 1

На рисунке 1 предоставлен код полученной программы:

```

1  #Задание 1
2  objects = [
3      ("Containment Cell A", 4),
4      ("Archive Vault", 1),
5      ("Bio Lab Sector", 3),
6      ("Observation Wing", 2)
7 ]
8  print(sorted(objects, key=lambda x: x[1]))

```

Рисунок 1 — Листинг программы для задания 1

Пояснение работы программы:

- 1) Создаём словарь со списком объектов с уровнем угрозы;
- 2) Сортируем с по возрастанию уровня угрозы с помощью команды sorted;
- 3) Делаем это коротко с помощью лямбда-выражения;
- 4) Выводим результат.

### 1.2 Задание 2

На рисунке 2 предоставлен код полученной программы:

```

10 #Задание 2
11 staff_shifts = [
12     {"name": "Dr. Shaw", "shift_cost": 120, "shifts": 15},
13     {"name": "Agent Torres", "shift_cost": 90, "shifts": 22},
14     {"name": "Researcher Hall", "shift_cost": 150, "shifts": 10}
15 ]
16 y = list(map(lambda x: x["shift_cost"] * x["shifts"], staff_shifts))
17 print(y)
18 print(max(y))

```

Рисунок 2 — Листинг программы для задания 2

Пояснение работы программы:

- 1) Создаём словарь со списком сотрудников, количеством и стоимостью их смен;
- 2) С помощью лямбда-выражения находим сколько сотрудник получил за смену;
- 3) С помощью команды map перебираем в словаре каждый элемент;
- 3) С помощью list создаём словарь;

- 4) Выводим результат;
- 5) С помощью команды `max` выводим максимальное из значений в списке.

### 1.3 Задание 3

На рисунке 3 предоставлен код полученной программы:

```

20  #Задание 3
21  personnel = [
22      {"name": "Dr. Klein", "clearance": 2},
23      {"name": "Agent Brooks", "clearance": 4},
24      {"name": "Technician Reed", "clearance": 1}
25 ]
26  new = list(map(lambda x: {"**x", "category": "Restricted" if x["clearance"] == 1
27                      else "Confidential" if 2 <= x["clearance"] <= 3
28                      else "Top Secret"}, personnel))
29  print(new)

```

Рисунок 3 — Листинг программы для задания 3

Пояснение работы программы:

- 1) Создаём словарь со списком с персоналом и их уровнем доступа;
- 2) Создаём новый словарь с дополнительным элементом;
- 3) С помощью лямбда-выражения создаём условие с названием для соответствующих уровней;
- 4) С помощью команды `map` применяем к каждому элементу словаря;
- 5) Выводим результат.

### 1.4 Задание 4

На рисунке 4 предоставлен код полученной программы:

```

31  #Задание 4
32  zones = [
33      {"zone": "Sector-12", "active_from": 8, "active_to": 18},
34      {"zone": "Deep Storage", "active_from": 0, "active_to": 24},
35      {"zone": "Research Wing", "active_from": 9, "active_to": 17}
36 ]
37  print(list(filter(lambda x: x["active_from"] <= 8 and x["active_to"] >= 18, zones)))

```

Рисунок 4 — Листинг программы для задания 4

Пояснение работы программы:

- 1) Создаём словарь со списком зон с временем активности (в часах);
- 2) С помощью лямбда-выражения ставим ограничения на выбор зон, которые работает в промежутке с 8 до 18 включительно;
- 3) Создаём из этого новый словарь;

- 4) С помощью команды filter вносим только подходящие элементы из исходного словаря;
- 5) Выводим результат.

## 1.5 Задание 5

На рисунке 5 предоставлен код полученной программы:

```

39 #Задание 5
40 import re
41 reports = [
42     {"author": "Dr. Moss", "text": "Analysis completed. Reference: http://external-archive.net"},
43     {"author": "Agent Lee", "text": "Incident resolved without escalation."},
44     {"author": "Dr. Patel", "text": "Supplementary data available at https://secure-research.org"},
45     {"author": "Supervisor Kane", "text": "No anomalies detected during inspection."},
46     {"author": "Researcher Bloom", "text": "Extended observations uploaded to http://research-notes.lab"},
47     {"author": "Agent Novak", "text": "Perimeter secured. No external interference observed."},
48     {"author": "Dr. Hargreeve", "text": "Full containment log stored at https://internal-db.scp"},
49     {"author": "Technician Moore", "text": "Routine maintenance completed successfully."},
50     {"author": "Dr. Alvarez", "text": "Cross-reference materials: http://crosslink.foundation"},
51     {"author": "Security Officer Tan", "text": "Shift completed without incidents."},
52     {"author": "Analyst Wright", "text": "Statistical model published at https://analysis-hub.org"},
53     {"author": "Dr. Kowalski", "text": "Behavioral deviations documented internally."},
54     {"author": "Agent Fischer", "text": "Additional footage archived: http://video-storage.sec"},
55     {"author": "Senior Researcher Hall", "text": "All test results verified and approved."},
56     {"author": "Operations Lead Grant", "text": "Emergency protocol draft shared via https://ops-share.scp"}
57 ]
58 new = list(filter(lambda x: x, [{**y, "text": re.sub(r'https?://\S+', '[ДАННЫЕ УДАЛЕНЫ]', y['text'])}
59 | | for y in reports]))
60 print(new)

```

Рисунок 5 — Листинг программы для задания 5

Пояснение работы программы:

- 1) Создаём словарь с анализом отчётов;
- 2) С помощью лямбда-выражения вместо ссылок http и https вписываем [ДАННЫЕ УДАЛЕНЫ];
- 3) С помощью команды filter выбираем подходящие элементы заменяем их;
- 4) С помощью list создаём новый словарь;
- 5) Выводим результат.

## 1.6 Задание 6

На рисунке 6 предоставлен код полученной программы:

```

62  #Задание 6
63  scp_objects = [
64      {"scp": "SCP-096", "class": "Euclid"},
65      {"scp": "SCP-173", "class": "Euclid"},
66      {"scp": "SCP-055", "class": "Keter"},
67      {"scp": "SCP-999", "class": "Safe"},
68      {"scp": "SCP-3001", "class": "Keter"}
69  ]
70  print(list(filter(lambda x: x["class"] != "Safe", scp_objects)))

```

Рисунок 6 — Листинг программы для задания 6

Пояснение работы программы:

- 1) Создаём словарь со списком SCP-объектов и их класса;
- 2) С помощью лямбда-выражения создаём условие "class не равен safe";
- 3) С помощью команды filter выбираем подходящие под условие элементы;
- 4) Создаём новый словарь;
- 5) Выводим результат.

## 1.7 Задание 7

На рисунке 7 предоставлен код полученной программы:

```

72  #Задание 7
73  incidents = [
74      {"id": 101, "staff": 4},
75      {"id": 102, "staff": 12},
76      {"id": 103, "staff": 7},
77      {"id": 104, "staff": 20}
78  ]
79  print(sorted(incidents, key=lambda x: -x["staff"])[0:3])

```

Рисунок 7 — Листинг программы для задания 7

Пояснение работы программы:

- 1) Создаём словарь со списком индексов и кол-ва задействованного персонала;
- 2) Создаём новый словарь;
- 3) С помощью команды sorted сортируем индекс по количеству персонала;
- 4) С помощью лямбда-выражения выбираем три первых элемента;
- 5) Выводим результат.

## 1.8 Задание 8

На рисунке 8 предоставлен код полученной программы:

```

82 #Задание 8
83 protocols = [
84     ("Lockdown", 5),
85     ("Evacuation", 4),
86     ("Data Wipe", 3),
87     ("Routine Scan", 1)
88 ]
89 print(list(map(lambda x: f"Protocol {x[0]} - Criticality {x[1]}", protocols)))

```

Рисунок 9 — Листинг программы для задания 8

Пояснение работы программы:

- 1) Создаём словарь со списком протоколов безопасности и их уровней критичности;
- 2) Создаём новый словарь с помощью list;
- 3) С помощью лямбда-выражения записываем через f"" нужное высказывание;
- 4) С помощью команды map применяем ко всем;
- 5) Выводим результат.

## 1.9 Задание 9

На рисунке 9 предоставлен код полученной программы:

```

91 #Задание 9
92 shifts = [6, 12, 8, 24, 10, 4]
93 print(list(filter(lambda x: 8 <= x <= 12, shifts)))

```

Рисунок 9 — Листинг программы для задания 9

Пояснение работы программы:

- 1) Создаём словарь со списком смен охраны и указанием длительности (в часах);
- 2) С помощью лямбда-выражения ставим ограничения на выбор смен, которые длились от 8 до 12 часов;
- 3) С помощью команды filter выбираем только подходящие под условие варианты;
- 4) Создаём словарь с помощью list;
- 5) Выводим результат.

## 1.10 Задание 10

На рисунке 10 предоставлен код полученной программы:

```
95  #Задание 10
96  evaluations = [
97      {"name": "Agent Cole", "score": 78},
98      {"name": "Dr. Weiss", "score": 92},
99      {"name": "Technician Moore", "score": 61},
100     {"name": "Researcher Lin", "score": 88}
101 ]
102 best = max(evaluations, key=lambda x: x["score"])
103 print(f"{best['name']} - {best['score']}")
```

Рисунок 10 — Листинг программы для задания 10

Пояснение работы программы:

- 1) Создаём словарь со списком сотрудников и результатами их психологического теста (от 0 до 100);
- 2) С помощью лямбда-выражения ищем по ключу "score" значение;
- 3) С помощью команды max выбираем наибольший результат;
- 4) Выводим результат и именем лучшего сотрудника.